Projet Programmation Java

Quiz test de connaissances

1) Contexte

Le but de ce projet est de permettre aux utilisateurs de tester leurs connaissances grâce à des quiz. Le logiciel développé pourrait être utilisé par des étudiants pour réviser leurs cours, sous une forme ludique. Il sera très général et paramétrable pour permettre des révisions sur des sujets divers. Mais l'objectif est qu'il soit utilisable l'année prochaine par les étudiants de première année du BUT Informatique, pour réviser leurs connaissances dans le langage Java. Dans le cadre des études à l'IUT, il pourra également servir à progresser en orthographe, avec des questions adaptées à cet usage.

Le projet lui-même sera développé dans le langage Java.

2) Les questions

Les questions seront réparties dans des catégories. Elles seront dotées aussi d'un niveau de difficulté : 1, 2, ou 3 pour les questions les plus difficiles. Dans tous les cas, une question sera constituée :

- ✓ d'un libellé
- ✓ d'une unique proposition de réponse juste
- ✓ d'au moins une proposition de réponse fausse et au plus 4
- ✓ et éventuellement d'un texte de feedback qui sera affiché en tant que corrigé de la question

Par exemple, dans la catégorie « Bases du langage Java», on pourrait trouver une question de niveau 1 dont le libellé serait

Quel est le type de l'expression suivante '9' ?

et les propositions de réponse : *String* *int* *char*

3) Gestion des catégories

L'application possèdera une catégorie nommée « Général ». L'utilisateur pourra en créer de nouvelles. On n'autorisera pas d'homonyme pour les noms de catégorie.

Il sera possible de modifier le nom d'une catégorie. De même la suppression d'une catégorie doit être autorisée. Toutefois, si celle-ci contient des questions, l'application en informera l'utilisateur qui pourra

confirmer ou pas la suppression demandée. La suppression de la catégorie entraîne celle des questions qu'elle contient.

Par contre, il ne sera pas possible de modifier le nom de la catégorie « Général », ni de la supprimer.

4) Gestion des questions

L'utilisateur aura la possibilité d'ajouter de nouvelles questions. L'interface devra être la plus ergonomique possible pour qu'il sélectionne facilement la catégorie et le niveau de la question.

Une fois la question créée, on doit pouvoir la modifier ou la supprimer. Les modifications incluent celles portant sur le libellé, les propositions de réponse et le déplacement dans une autre catégorie, ainsi que le changement du niveau de difficulté, ou la réécriture du feedback.

La saisie des questions est une tâche fastidieuse. On souhaite proposer des facilités pour les intégrer à l'application. L'idée est la suivante. On pourra préparer au préalable une liste de questions dans un outil bureautique de type tableur (Excel par exemple). Les questions seront présentées dans un format bien précis : une question par ligne, et les colonnes préciseront dans l'ordre :

- ✓ La catégorie
- ✓ Le niveau
- ✓ Le libellé de la question
- ✓ La proposition juste
- ✓ Une proposition fausse
- ✓ éventuellement sur les colonnes suivantes d'autres propositions fausses (au plus 4, au total)
- ✓ éventuellement un texte de feedback qui sera affiché en même temps que le corrigé d'un questionnaire (voir plus loin)

Ce tableau ainsi constitué sera enregistré au format *csv*. L'application devra donc être capable d'intégrer le contenu d'un tel fichier dans la banque de questions. Bien sûr, il faudra envisager le cas où le format du fichier serait incorrect. Un message d'erreur précis informera de l'échec de l'importation du fichier, ou de certaines lignes, et expliquera la cause.

Si le fichier contient des catégories qui n'existent pas encore au sein de l'application, elles seront créées automatiquement. Eventuellement, l'utilisateur sera informé de la création de ces nouvelles catégories. S'il existe déjà une question identique (même catégorie, même libellé et mêmes propositions de réponse, donc le niveau n'intervient pas, de même que le feedback), la question ne sera pas importée. L'utilisateur sera informé.

5) Lancement d'un questionnaire par un joueur

Avant de lancer un questionnaire, le joueur devra au préalable choisir le type de celui-ci en précisant :

- ✓ Un niveau de difficulté, éventuellement celui-ci pourra être « indifférent »
- ✓ Une catégorie de questions. Le joueur pourra ne pas faire ce choix. Les questions seront alors choisies dans l'ensemble des catégories.
- ✓ Le nombre de questions auxquelles il souhaite répondre : 5, 10, ou 20.

Une fois ces paramètres spécifiés, l'application affichera les questions choisies au hasard parmi celles de la banque de questions. Notons que les propositions de réponses elles aussi seront affichées dans un ordre aléatoire.

Pour toute question, le joueur est autorisé à ne pas se prononcer quant au choix de la proposition correcte. Il peut dans ce cas passer à la question suivante. Mais bien sûr la non-réponse à la question sera comptabilisée parmi les réponses incorrectes. Une fois le questionnaire validé, l'application affichera le

résultat sous la forme d'un pourcentage de réponses justes et d'une phrase adaptée au score obtenu : « Félicitations ! Toutes vos réponses sont justes ! », « Bravo ! plus de la moitié de vos réponses sont justes ! » … Elle proposera également au joueur de visualiser les réponses correctes. C'est lors de la visualisation des bonnes réponses que le feedback sera affiché, s'il a été renseigné.

Il faudra envisager le cas où la banque des questions ne contiendrait pas suffisamment de questions ayant les propriétés choisies par le joueur. Dans ce cas, un message l'informera de cette particularité, lui indiquera le nombre de questions que l'application est en mesure de lui soumettre, et lui demandera s'il souhaite poursuivre. Bien sûr, cette dernière demande ne doit être faite que s'il y a au moins une question répondant aux critères.

6) Communication entre deux ordinateurs

Il sera possible d'exporter l'ensemble des questions (avec la catégorie) gérées par l'application dans un fichier, au format *csv* par exemple. Pour ce faire l'application devra au préalable se relier à un ordinateur distant jouant le rôle de client, et sur lequel l'application Quiz aura été lancée. Une fois la connexion entre les deux ordinateurs établie, les questions de la banque de question seront envoyées de manière cryptée vers la machine distante.

Le fichier envoyé sera crypté en utilisant un algorithme de chiffrement symétrique : le chiffrement de Vigenère. Dans un premier temps, le partage de la clé de chiffrement se fera directement « en clair », par exemple par l'envoi d'un fichier au format texte contenant la clé à l'ordinateur distant. Dans un deuxième temps, l'échange de Diffie-Hellman permettra aux deux ordinateurs distants de partager une donnée secrète. Et la clé de chiffrement sera construite à partir de cette donnée secrète.

L'utilisateur doit avoir la possibilité d'envoyer soit toutes les questions de la banque de questions, soit uniquement celles qu'il aura sélectionnées. Il pourra sélectionner au choix une ou plusieurs catégories, ou encore individuellement certaines questions.

Sur la machine cliente, lors de l'importation, des vérifications devront être faites : format de la question, éventuellement création d'une nouvelle catégorie, vérification qu'une question identique n'existe pas déjà. Pour ces vérifications, le comportement de l'application sera similaire à celui adopté lors de l'importation d'un fichier *csv* local (voir dans les paragraphes précédents).

Précisions

On suppose que :
- l'échange de fichier n'est possible qu'entre 2 appareils à la fois
- les 2 appareils sont préalablement intégrés à un réseau (en particulier, disposent d'une IP)
- l'application dispose d'un bouton permettant d'afficher l'IP de l'appareil

Contraintes techniques

- utiliser des sockets Java en mode connecté avec les classes Socket et ServerSocket
- une connexion par échange de fichier, arrêtée automatiquement à la fin de l'échange

7) Autres spécifications

L'utilisateur aura la possibilité de visualiser une notice d'utilisation de l'application. Celle-ci expliquera notamment le format du fichier Excel permettant l'importation de questions.

Via les paramètres, l'utilisateur pourra renseigner son nom. Celui-ci permettra de personnaliser les messages informant des résultats, notamment.

L'application sera livrée avec une banque de questions initiales que aurez conçue et d'autres fournies par l'enseignante (certaines porteront sur le langage Java, d'autres sur l'orthographe).

La persistance des données (donc de la banque de questions) sera assurée grâce à la sérialisation. Si nécessaire, un support de cours vous sera donné. Il n'est donc pas demandé de mettre ne place une base de données.

L'interface graphique devra être aussi ergonomique que possible. On gèrera au mieux les erreurs ou anomalies qui pourraient se produire.

8) Livrables attendus

8.1) Livrable « Développement » (Compétence 1 – Réaliser – AC1 et AC3)

Les livrables sont les suivants :

- tout le code source de l'application, y compris les programmes de test
- un exécutable de l'application
- un dossier technique au format *pdf*

Le dossier technique au format pdf comportera :

- une page de garde
- un sommaire
- une introduction : résumé de l'application réalisée (une demi-page au moins et jusqu'à une page)
- un diagramme de cas d'utilisation UML accompagné de :
    - soit la description écrite des cas d'utilisation
    - soit des User Stories
- la conception de l'application avec un diagramme de classes UML et des explications
- une présentation des tests unitaires réalisés. Pour chaque test unitaire, il faut indiquer :
    - le nom de la classe testée et son paquetage
    - le nom de la classe de test unitaire et son paquetage
    - des explications qui justifient que les jeux de test et les algorithmes de tests choisis pour cette classe sont pertinents
- les tests d'intégration effectués ou bien les jeux d'essai utilisés pour tester l'application finale : quelles opérations vous avez effectuées pour tester, quel a été le résultat, dans les conditions normales d'utilisation et aussi dans les conditions anormales. Vous pouvez utiliser des copies d'écran accompagnées d'explications. Si certains cas testés ont conduit à un résultat identique, il n'est pas utile de mettre une copie d'écran pour chacun d'eux. Vous indiquerez toutefois quels sont les cas qui ont conduit à un résultat similaire à la copie d'écran. Le but est de démontrer la bonne qualité de votre couverture de tests.
- un bilan faisant état des difficultés rencontrées et des solutions apportées
- une conclusion pour **chacun** des étudiants du groupe et éventuellement une conclusion globale au groupe

La qualité de la rédaction (clarté, précision, mise en forme, orthographe) sera prise en compte dans l'évaluation du projet.

8.2) Livrable « chiffrement » (Compétence 3 – Administrer – AC1)

Un document au format pdf qui doit au moins contenir :

- Une introduction.
- Toutes les précisions nécessaires sur l'algorithme de chiffrement de Vigenère effectivement mis en œuvre. En particulier :
    - Quels sont les documents qu'il est possible de chiffrer (alphabet utilisé) ?
    - Comment les « calculs modulaires » sur l'alphabet sont-ils conduits ?
    - Comment la clé de chiffrement est-elle construite et partagée (sans, puis avec l'échange de Diffie-Hellman) ?
- Un schéma, le plus clair possible, présentant les différentes étapes nécessaires au chiffrement/déchiffrement des données (côté ordinateur qui exporte les données et côté ordinateur distant).
- Toutes réflexions que vous jugerez utiles.
- Un bilan individuel, pour chaque membre du groupe, spécifique à la partie chiffrement.
- Les codes sources, bien commentés, liés au chiffrement : choix de la clé, chiffrement et déchiffrement.

La bonne forme de ce document sera prise en compte.

8.3) Livrable « réseau » (Compétence 3 – Administrer – AC1)

Un document en PDF contenant les points suivants :

- préciser et expliquer votre choix duquel appareil est serveur, duquel est client
- un justificatif du choix du numéro de port du serveur
- un diagramme de séquences pour chaque type d'échange permis par l'application (fichier contenant la clé, donnée secrète, fichier de questions, etc), ainsi que le format des données échangées
- le code source des classes intégrant des échanges réseau, avec explications